Discrete Applied Mathematics 161 (2013) 28–42

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Batching and delivery in semi-online distribution systems

Igor Averbakh *, Mehmet Baysan

Department of Management, University of Toronto Scarborough, 1265 Military Trail, Toronto, Ontario M1C 1A4, Canada

ARTICLE INFO

Article history:
Received 12 December 2010
Received in revised form 21 May 2012
Accepted 3 August 2012
Available online 1 September 2012

Keywords:
Supply chain scheduling
Semi-online algorithm
Competitive analysis

ABSTRACT

Suppose a distribution center receives orders from customers for delivery of some goods. Orders may be delayed to be grouped and delivered in batches. The cost of one delivery does not depend on the number of orders taken. The total cost (to be minimized) is the sum of the total delay time of the orders and the total delivery cost. In the on-line environment, where at any time there is no information about future orders, there is a simple on-line algorithm with competitive ratio 2 which is known to be the best possible. We consider the semi-online environment where at any instant we know the orders that will be released in the next S time units, but have no information about the orders that will be released later. For this environment, we present a semi-online algorithm and analyze its competitive ratio.

1. Introduction

1.1. Problem statement

Suppose that there is a distribution center (e.g., a warehouse) that receives orders from customers for delivery of some goods or product, and performs the deliveries. Orders $o_1, \ldots, o_n$ are released by customers at release times $r_1, \ldots, r_n$, respectively, $0 < r_1 \leq r_2 \leq \cdots \leq r_n$. For convenience, we will also define $r_0 = 0$ and $r_{n+1} = +\infty$. We assume that any order is received by the distribution center as soon as it is released by the corresponding customer, so in the paper, "order released" means also "order received by the distribution center". Processing time of an order is assumed to be 0 (negligible), thus, an order can be delivered as soon as it is released. To minimize costs, some orders may not be delivered as soon as they are released, and may be delayed so as to deliver them in batches. We assume that a delivery can combine orders of different customers, can include any number of orders, takes zero time, and its cost D is fixed and does not depend on the number of orders included. The *delay time* of an order is the time between its release and delivery to the customer. The *total delay time* is the sum of the delay times of all orders. The problem is to schedule the deliveries so as to minimize the *total cost*, which is the sum of the total cost of all deliveries and the total delay time of the orders. That is, if p deliveries are scheduled, and $d(o_i)$ is the time of the delivery that includes order o_i, then the total cost is $pD + \sum_{i=1}^{n}(d(o_i) - r_i)$.

Our assumption that a delivery can combine orders of different customers represents the situation where customers are located at the same place. If there are several groups of customers located at different places, and orders from different groups cannot be combined for delivery (the case of *direct shipping*, see [6]), then the problem decomposes into independent subproblems for the groups, and the results of this work are still applicable. However, if deliveries to groups located at different places can be combined (the case of batch delivery with routing, see [6]), and there is flexibility with choosing delivery routes for delivery vehicles, then additional routing issues arise; this case is not considered in this paper.

Note that the objective combines cost and time. It is typical in such problems to assume that delaying orders incurs cost; a common assumption is that the cost incurred by delaying an order is proportional to the delay time, and delay time can

* Corresponding author. Fax: +1 416 287 7392.
E-mail addresses: averbakh@utsc.utoronto.ca (I. Averbakh), m.baysan@utoronto.ca (M. Baysan).

0166-218X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2012.08.003

be converted to delay cost by appropriate scaling and changing the units. We assume that such scaling has been done, and will interpret delay time as the delay cost; thus, we assume that time and cost are measured in the same units, and we will use the terms "delay time" and "delay cost" interchangeably.

In the off-line environment, where all future orders are known in advance, an optimal schedule can be obtained quickly (in $O(n^2)$ time) by standard dynamic programming (see Lemma 1 and its proof in the Appendix). In the on-line environment, where at any time there is no information about future orders, there is a simple on-line algorithm with complexity $O(n)$ and competitive ratio 2 [4]; that is, the algorithm obtains a solution with a total cost which can be at most twice the total cost of an optimal off-line solution. Moreover, the results of [4] imply that this competitive ratio cannot be improved, that is, there is no on-line algorithm with a competitive ratio less than 2. The purpose of this paper is to consider the semi-online case, where future is known for the next S time units; that is, at any instant τ we know the orders that will be released in the time interval $[\tau, \tau + S]$ but have no information about orders that will be released after the instant $\tau + S$, which is called the *visibility horizon*. This problem will be referred to as **Problem P**$^{(S)}$. Then, notation Problem $P^{(\infty)}$, or simply *Problem P*, naturally corresponds to the off-line version, and Problem $P^{(0)}$ corresponds to the on-line version. We would like to obtain a semi-online algorithm that utilizes the partial information about the future to achieve a better competitive ratio. Intuitively, it is clear that the possibilities for improving the competitive ratio with respect to the best possible on-line competitive ratio of 2 depend on the value of S; more precisely, on the ratio S/D. When S/D is close to 0, we have an almost on-line case, and thus it should not be possible to obtain a competitive ratio significantly better than 2. As S/D grows, more information about the future can be taken into account, and thus there are more possibilities for obtaining a better competitive ratio. When S/D is a large number, we have an almost off-line case, and thus it should be possible to obtain a competitive ratio close to 1. Thus, the competitive ratio of our semi-online algorithm should be a function of S and D.

The main contribution of the paper is a semi-online algorithm for Problem $P^{(S)}$ with a competitive ratio ρ where

$$\rho = \begin{cases} \dfrac{2D + 0.5S}{D + 0.5S} & \text{when } 0 < S < 2D, \\ \dfrac{S + D}{S} & \text{when } S \geq 2D. \end{cases}$$

Our semi-online algorithm consists of two algorithms, Algorithms 1 and 2; the former is designed for the case $0 < S < 2D$, and the latter for the case $S \geq 2D$. Hence there are two expressions for the competitive ratio depending on the case; the point $S = 2D$ separates the areas of applicability of Algorithms 1 and 2. Algorithm 1 can be implemented with computational complexity $O(n^2 \log n)$, and Algorithm 2 can be implemented with computational complexity $O(n^2)$.

Note that the competitive ratio ρ has the expected properties with respect to the values of the parameters S and D. It is close to 2 when S/D is close to zero, and monotonically decreases as S/D increases. Finally, it approaches 1 when $S/D \to \infty$.

1.2. Motivation and related problems

The semi-online Problem $P^{(S)}$ may be relevant, for example, in the following fully on-line situation. Suppose that the future is unknown, but each order can be delivered not earlier than S' time units after its release due to, e.g., time required for registration, assembly, etc. In other words, all orders have processing time S', but can be processed in parallel, so processing of each order starts as soon as the order is released. Then, it would be natural to define the delay time of each order as the time between its release and delivery minus S'. The on-line problem of minimizing the sum of the total delay time plus the total cost of deliveries in this situation is equivalent to Problem $P^{(S')}$, because at any instant τ, we know all orders that will become available for delivery in the interval $[\tau, \tau + S']$ (these are the orders that were released during $[\tau - S', \tau]$). Problem $P^{(S')}$ would also be relevant if in this situation the processing times of orders may be different but cannot be smaller than S'. (We note that this context and the problem are fundamentally different from those considered in [3]; in [3], one of the considered semi-online environments was also defined by information about a lower bound on processing times of all future orders, but the orders had to be processed sequentially, so there was a fixed minimum distance between completion times of different orders which played a critical role in the analysis.)

Let us briefly review some related work. In [4], a problem similar to Problem $P^{(0)}$ was studied in a purely on-line environment, where additionally it was assumed that the orders (interpreted as jobs) may have non-zero processing times, no more than one order can be processed simultaneously, and preemption is allowed. The effects of capacitated deliveries for this model in the purely on-line environment were studied in [2]. Some semi-online environments defined by partial information about processing times of all future jobs and the "density" of the instance (where density is the ratio of the total processing time of all jobs to a suitably defined "time length" of the instance) were studied in [3].

Problem $P^{(S)}$ belongs to the general class of supply chain scheduling problems [9,7] where it is required to co-ordinate production, batching and delivery decisions between several levels of a supply chain, and to the more narrow class of production–distribution problems [6] that is focused on two-level supply chains. The literature on off-line supply chain scheduling and related problems is quite extensive; we refer the reader to the recent survey [6] on production–distribution problems, and to the bibliographies in [7,9,10]. As for related on-line models, a non-preemptive on-line problem where each job is delivered individually, there is a transportation time for job delivery, and the objective is to minimize the maximum delivery time of the jobs, was considered in [13,11,16]. On-line scheduling on a single batch machine where several jobs

can be processed simultaneously as a batch, the processing time of a batch is the longest processing time of its jobs, the jobs should be subsequently delivered to their destinations, and the objective is to minimize the time by which all jobs have been delivered, was studied in [15]. A similar on-line batch scheduling problem with restarts, where a running batch may be interrupted losing all the work done on it, and without delivery considerations, was investigated in [8].

The structure of the paper is as follows. In Section 2, we introduce auxiliary notation and definitions. The algorithm for the case $0 < S < 2D$ is presented and analyzed in Section 3. Section 4 is devoted to the algorithm for the case $S \geq 2D$. A lower bound on the competitive ratio of any deterministic semi-online algorithm for Problem $P^{(S)}$ is presented in Section 5. Some final remarks are made in Section 6.

2. Preliminaries

For any real numbers $a, b, a \leq b$, let $[a, b]$ ((a, b)) denote the interval between a and b including (not including) the endpoints. For any integer k, m, let $[k : m]$ be the set $\{k, k+1, \ldots, m\}$ if $k \leq m$ and $\emptyset$ if $k > m$. At any instant t, an order that was released earlier than t but was not delivered before t is called *waiting*. Note that an order that is released at t cannot be waiting at t. An order that is waiting at t or that is released at t is called *available* at t (even if a delivery happens at t that delivers the order). For any order and an interval (a, b) and a specific schedule of deliveries, the order's *delay time* in (a, b) is the time that the order spends waiting in (a, b). For an order with release time r and an interval (a, b), the *uninterrupted delay time of the order in* (a, b) is defined as $b - \min\{\max\{a, r\}, b\}$; it is the delay time of the order in (a, b) assuming that the order is not delivered before b. Note that the uninterrupted delay time in (a, b) can be different from the delay time in (a, b), if the order is in fact delivered before b.

A solution X to Problem P can be represented as a vector of consecutive delivery times, $X = (d_1, \ldots, d_p)$, $d_1 < \cdots < d_p$, $p \leq n$. It is clear that in an optimal solution to Problem P, each delivery is at a release time of some order and takes all available orders, and there is a delivery at r_n. Let $X' = (x'_1, \ldots, x'_{p'})$ and $X'' = (x''_1, \ldots, x''_{p''})$ be two optimal solutions for Problem P. Then, X' is called *lexicographically smaller* than X'' if either $\{x''_1, \ldots, x''_{p''}\}$ is a proper subset of $\{x'_1, \ldots, x'_{p'}\}$ or there is $k \in [1 : p']$ such that $x'_i = x''_i$ for $i \in [1 : k-1]$ and $x'_k < x''_k$.

Lemma 1. *There is an optimal solution to Problem P that is lexicographically smaller than any other optimal solution to the problem; this solution can be found in $O(n^2)$ time.*

Proof. See Appendix. □

For any instants a, b, $0 \leq a < b$, let $P(a, b)$ be the (off-line) problem of finding a delivery schedule in $[a, b)$ to minimize the sum of the total cost of deliveries in $[a, b)$ and the total delay time in (a, b) of all orders that are released during $[a, b)$. Note that in $P(a, b)$ we take into account only the orders that are released in $[a, b)$; also, we do not take into account the delay time incurred outside of (a, b), so in an optimal solution to $P(a, b)$ there may be some orders released in $[a, b)$ after the last delivery (they remain undelivered). Let $Z(a, b)$ be the optimal objective value of $P(a, b)$, and let $O(a, b)$ be the lexicographically smallest optimal schedule for $P(a, b)$; it can be found using a straightforward modification of the algorithm in the proof of Lemma 1. At this point we assume that $Z(a, b)$ and $O(a, b)$ can be available at any instant when all arrivals in $[a, b)$ are known; we will further elaborate on this assumption when we discuss implementation issues.

An on-line algorithm for a minimization problem with a nonnegative objective is called *ρ-competitive*, for some $\rho \geq 1$, if for any problem instance the objective value of the solution obtained by the algorithm is bounded by ρ times the optimal off-line objective value for the instance plus a constant [5,12]. (Since the scheduling problems that we consider are scalable, the additive constant can be deleted from the definition.) The *competitive ratio* of an algorithm is the infimum of the set of all values ρ such that the algorithm is ρ-competitive.

3. The case $0 < S < 2D$

3.1. The algorithm

Assume $0 < S < 2D$. The following algorithm is proposed in this case for Problem $P^{(S)}$.

Algorithm 1. Initially, set $k = 1, A^1 = 0$. Value k will be interpreted as a stage number. Let CT be the current time (initially 0).

Step 1. Wait until $O(A^k, CT + S)$ includes a delivery before or at CT. Suppose that this happens when $CT = E^k$ for some E^k.

Step 2. If $Z(A^k, E^k + S) < D + 0.5S$ then make a delivery at E^k; wait until $CT = E^k + 0.5S$; set $A^{k+1} = E^k + 0.5S$, $k := k + 1$, and go to Step 1.
Otherwise, go to Step 3.

Step 3. ($Z(A^k, E^k + S) \geq D + 0.5S$.) Solve the off-line problem of finding a delivery schedule in $[E^k, E^k + S)$ to minimize the sum of the total cost of deliveries in $[E^k, E^k + S)$ and the total delay time in $[E^k, E^k + S)$ of all orders that are waiting at E^k or are released during $[E^k, E^k + S)$, subject to the condition that there must be a delivery immediately after the last order release in $[E^k, E^k + S)$, or at E^k if there are no order releases in $[E^k, E^k + S)$. We call this problem $P'(k)$; let $Z'(k)$ and $O'(k)$ denote its optimal objective value and optimal solution, respectively. ($P'(k)$ is of the same type as Problem P.) Schedule deliveries in $[E^k, E^k + S)$ according to $O'(k)$. Wait until $CT = E^k + S$. Set $A^{k+1} = E^k + S$, $k := k + 1$, and go to Step 1.

The description of the algorithm is completed.

Eventually, the algorithm waits indefinitely at Step 1 without having any available orders and without scheduling any deliveries. To avoid this formal inconvenience, we can assume that an upper bound is known for the release time of the last order, and we stop when CT is greater than this upper bound and there are no available orders.

Observe that the number of stages cannot exceed $n + 1$, because at each stage except the last one, at least one order is delivered.

Instants E^k will be called *trigger instants* for Algorithm 1.

The following observation is important for understanding Step 2.

Observation 1. If $Z(A^k, E^k + S) < D + 0.5S$, then no orders are released in $(E^k, E^k + 0.5S)$.

Proof. In $O(A^k, E^k + S)$, there is a delivery in $[A^k, E^k]$; since $Z(A^k, E^k + S) < D + 0.5S$, and $S < 2D$, there are no other deliveries in $[A^k, E^k + S)$. Now, if there is an available order in $(E^k, E^k + 0.5S)$, then this order will be waiting in $(E^k + 0.5S, E^k + S)$, and therefore $Z(A^k, E^k + S) \geq D + 0.5S$. Contradiction. □

To illustrate the action of Algorithm 1, we consider two small examples.

Example 1. *There is only one order released at time* $r_1 = 1$; $S = 4$, $D = 7$. Initially, $k = 1$, $A^1 = 0$. Observe that $O(0, CT + 4)$ does not have any deliveries if $CT < 4$. The first trigger instant is $E^1 = 4$ when $O(0, 8)$ has a delivery at time 1. Since $Z(0, 8) = 7$ is less than $D + 0.5S = 9$, Step 2 schedules a delivery at $E^1 = 4$, waits until $CT = 6$, sets $A^2 = 6$, $k := 2$, and returns to Step 1. No more deliveries will be scheduled; the obtained solution (one delivery at 4) has objective value 10, and the approximation ratio is 10/7 since the optimal objective value is 7 (one delivery at time 1).

Example 2. *The same as* Example 1 *but* $S = 7$, $D = 4$. Initially, $k = 1$, $A^1 = 0$. Observe that $O(0, CT + 7)$ has one delivery at time 1 for any $CT \geq 0$. The first trigger instant is $E^1 = 1$. Since $Z(0, 8) = 4$ is less than $D + 0.5S = 7.5$, Step 2 schedules a delivery at $E^1 = 1$, waits until $CT = 4.5$, sets $A^2 = 4.5$, $k := 2$ and transfers to Step 1. No more deliveries will be scheduled; the obtained solution (one delivery at time 1) has objective value 4 and is, in fact, an optimal solution.

The following theorem is the main result of this section.

Theorem 1. *The competitive ratio of Algorithm 1 is* $\frac{2D+0.5S}{D+0.5S}$.

The next two subsections are devoted to proving Theorem 1, which will require a number of additional results.

3.2. Additional definitions and a sketch of the proof's framework

In this subsection, we provide additional definitions needed for the proof of Theorem 1, and give some insights and an informal sketch of the proof's framework. Let I_k denote the interval $[A^k, A^{k+1})$; it will be called *the k-th basic interval*. Let $I_{k_{\max}}$ be the last basic interval (i.e., $A^{k_{\max}} \leq r_n$ and $A^{k_{\max}+1} > r_n$). Let $ALG1$ be the solution obtained by Algorithm 1, and let OPT be the lexicographically smallest optimal off-line solution for Problem P. $ALG1$ and OPT will represent not only the schedules of deliveries but also full schedules that include orders' waiting. Let C_{ALG1} and C_{OPT} denote the total costs of $ALG1$ and OPT, respectively. For any interval I, let $C_{ALG1}(I)$ and $C_{OPT}(I)$ denote the total costs of $ALG1$ and OPT, respectively, over I (that is, the sum of the cost of deliveries in I and the delay time incurred inside I), and let $J(I)$ be the set of orders that are released in I. The value $\frac{C_{ALG1}(I)}{C_{OPT}(I)}$ is called the *ratio* for I.

Lemma 2. *For any* I_k, *in* $ALG1$ *there are no waiting orders at* A^{k+1}.

Proof. The statement follows from the description of the algorithm and Observation 1. □

Clearly, for any fixed instant a, $O(a, b)$ changes only at a finite number of values $b > a$ as b grows. Let $\hat{O}(k)$ be the schedule of deliveries in $O(A^k, CT + S)$ for values of CT immediately prior to E^k if $E^k \neq A^k$, and $\hat{O}(k) = O(A^k, E^k + S)$ if $E^k = A^k$.

Lemma 3. $\hat{O}(k)$ *is an optimal solution to problem* $P(A^k, E^k + S)$, *and all deliveries in* $\hat{O}(k)$ *happen not earlier than* E^k.

Proof. If $E^k = A^k$, the statement is obvious. If $E^k > A^k$, the statement follows from the definition of E^k and the observation that $Z(a, b)$ is a continuous function of b. □

We note that although $\hat{O}(k)$ is an optimal solution to $P(A^k, E^k+S)$, it may be different from the lexicographically smallest optimal solution $O(A^k, E^k+S)$ if $E^k > A^k$.

Definition. A basic interval I_k is called *end-incomplete* if there is a waiting order at A^{k+1} in *OPT*; otherwise it is called *end-complete*. Clearly $I_{k_{\max}}$ is end-complete. A basic interval I_k is called *start-incomplete* if there is a waiting order at A^k in *OPT*; otherwise it is called *start-complete*. Clearly, I_1 is start-complete.

We define a partition of $I_1 \cup \cdots \cup I_{k_{\max}}$ into *blocks*, where each block is the union of one or several consecutive basic intervals, using the following procedure.

Procedure P. $k := 1$.
Until $k > k_{\max}$ do.
If I_k is end-complete, make a block that consists of only I_k; set $k := k+1$.
Otherwise, make a block $I_k \cup I_{k+1} \cup \cdots \cup I_{k+p}$, where I_{k+p} is the first end-complete basic interval after I_k; set $k := k+p+1$.
End of procedure.

A block is called *nontrivial* if it consists of more than one basic interval; otherwise it is called *trivial*.

Observation 2. The first basic interval of any block is start-complete; all other basic intervals of the block are start-incomplete. The last basic interval of any block is end-complete; all other basic intervals of the block are end-incomplete.

Let us now outline some insights and an informal framework of the proof of Theorem 1 that will be given in the next subsection. Algorithm 1 makes scheduling decisions at trigger instants. Each trigger instant E^k has the following property: For the off-line problem $P(A^k, E^k+S)$ of minimizing the total cost incurred within the visibility horizon E^k+S by the currently undelivered known orders, there is an optimal solution $O(A^k, E^k+S)$ which has a delivery at or before E^k, and a (not necessarily different) optimal solution $\hat{O}(k)$ where all deliveries (if any exist in $\hat{O}(k)$) are not earlier than E^k. This property is heavily used in the analysis. Let us give some intuition on why this property is useful. Existence of the solution $\hat{O}(k)$ means that at the trigger instant E^k, we still can schedule deliveries of all currently available and already known future orders so that the total delay and delivery cost incurred by them is not greater than $Z(A^k, E^k+S)+D$ (e.g., by taking the solution $\hat{O}(k)$ and adding a delivery anywhere between the last already known order release and E^k+S). Since $Z(A^k, E^k+S)$ is a lower bound on $C_{OPT}([A^k, E^k+S))$, this allows us to obtain an effective local scheduling in $[A^k, E^k+S)$ if $Z(A^k, E^k+S)$ is sufficiently large. Existence of the solution $O(A^k, E^k+S)$ is used in different ways in proofs in the next subsection, but in general it means that some already released orders incur substantial delay costs if not delivered timely, and (along with the existence of $\hat{O}(k)$) helps to estimate the (local) costs of ALG1 and OPT with respect to each other.

Considering the specific rules of Algorithm 1, if $Z(A^k, E^k+S) \geq D+0.5S$, then defining $A^{k+1} = E^k+S$ and scheduling deliveries in $I_k = [A^k, A^{k+1})$ as done in Step 3 ensures that the ratio for I_k is not greater than $\frac{2D+0.5S}{D+0.5S}$; to prove this, we will use existence of the solution $\hat{O}(k)$ for $P(A^k, E^k+S)$. If $Z(A^k, E^k+S) < D+0.5S$, which is a much more complicated case for analysis, we will show that defining $A^k = E^k+0.5S$ and making a delivery at E^k as done in Step 2 ensures that $\frac{C_{ALG1}(I_k)}{C_{OPT}(I_k)} > \frac{2D+0.5S}{D+0.5S}$ is possible only if I_k is both end-incomplete and start-complete. This means that the ratio for any trivial block is not greater than $\frac{2D+0.5S}{D+0.5S}$, and for any non-trivial block B, for each basic interval I of the block except the first one $\frac{C_{ALG1}(I)}{C_{OPT}(I)} \leq \frac{2D+0.5S}{D+0.5S}$, and only the first basic interval of B can have ratio greater than $\frac{2D+0.5S}{D+0.5S}$. However, careful analysis of all possible situations will show that if this happens, then $\frac{C_{ALG1}(I')+C_{ALG1}(I'')}{C_{OPT}(I')+C_{OPT}(I'')} \leq \frac{2D+0.5S}{D+0.5S}$, where I' and I'' are the first and the last basic intervals of the block B, respectively. This implies that the ratio for B is not greater than $\frac{2D+0.5S}{D+0.5S}$. Existence of solutions $O(A^k, E^k+S)$ and $\hat{O}(k)$ for $P(A^k, E^k+S)$ is the basis of the analysis. Thus, the ratio for any block is not greater than $\frac{2D+0.5S}{D+0.5S}$, which implies that $\frac{C_{ALG1}}{C_{OPT}} \leq \frac{2D+0.5S}{D+0.5S}$. An example will show that $\frac{C_{ALG1}}{C_{OPT}}$ can be made arbitrarily close to $\frac{2D+0.5S}{D+0.5S}$, thus the bound is tight.

Definition. (a) We say that a basic interval I_k belongs to Category *A*, if in *ALG*1 deliveries in I_k are scheduled in Step 3 of Algorithm 1.
(b) We say that a basic interval I_k belongs to Category *B*, if in *ALG*1 the delivery in I_k is scheduled in Step 2 of Algorithm 1, and I_k is start-complete.
(c) We say that a basic interval I_k belongs to Category *C*, if in *ALG*1 the delivery in I_k is scheduled in Step 2 of Algorithm 1, and I_k is start-incomplete.

3.3. Analysis of Algorithm 1

Lemma 4. *If I_k belongs to Category A, then:*
(a) $C_{OPT}(I_k) \geq Z(A^k, E^k+S)$.
(b) $C_{ALG1}(I_k) \leq Z(A^k, E^k+S)+D$.
(c) $C_{ALG1}(I_k) \leq C_{OPT}(I_k)+D$.

Fig. 1. Illustration for the proof of Lemma 5.

Proof. Part (a) follows from the definitions. To prove part (b), we note that by taking the schedule $\hat{O}(k)$ and adding a delivery immediately after the last order release in $[E^k, E^k+S)$, or at E^k if there are no order releases in $[E^k, E^k+S)$, we get a schedule feasible to problem $P'(k)$ with total cost not greater than $Z(A^k, E^k + S) + D$ (see Lemma 3). Part (b) follows immediately. Part (c) follows from parts (a) and (b). □

Lemma 5. *If I_k belongs to Category A, and is end-complete, then:*

(a) *If in OPT there are at least two deliveries in I_k, then $C_{OPT}(I_k) \geq 2D + \delta$ and $C_{ALG1}(I_k) \leq 3D + \delta$ for some $\delta \geq 0$.*
(b) *If in OPT there are no deliveries in $[A^k, E^k)$, then $C_{OPT}(I_k) \geq C_{ALG1}(I_k) \geq D + 0.5S$.*
(c) *If in OPT there is one delivery in $[A^k, E^k)$ and there are no deliveries in $[E^k, A^{k+1})$, then $C_{OPT}(I_k) \geq D + 0.5S + \delta$, $C_{ALG1}(I_k) \leq 2D - 1.5S + \delta$ for some $\delta \geq 0$.*

Proof (*See* Fig. 1)**.** Part (a) is straightforward, taking into account Lemma 4(c).

Since I_k is end-complete, *OPT* must have a delivery immediately after the last order release in I_k. If the conditions of part (b) and the lemma hold, then the schedule of deliveries for *OPT* over $[E^k, E^k + S)$ must be a feasible solution to problem $P'(k)$, which (along with the definition of Category *A*) implies part (b).

Let us prove part (c). Suppose that the conditions of part (c) and the lemma hold. Let τ be the time of the last order release in I_k. Since I_k is end-complete, there must be a delivery at τ in *OPT*; this implies $A^k \leq \tau < E^k$, so there are no order releases in $[E^k, E^k + S)$. Since $C_{OPT}(I_k) \geq Z(A^k, E^k + S) \geq D + 0.5S$, the total delay time F of the orders that were released in $[A^k, \tau)$ in *OPT* is at least $0.5S$. Thus, there are at least two order releases in $[A^k, E^k)$.

Since in $\hat{O}(k)$ there are no deliveries in $[A^k, E^k)$ and since there are no order releases in $[E^k, A^{k+1})$, $\hat{O}(k)$ does not have deliveries at all. Let x be the number of order releases in I_k, $x \geq 2$ as noted above. Since I_k belongs to Category *A*, $Z(A^k, E^k+S) = D+0.5S+\delta$ for some $\delta \geq 0$, and this is the total delay time over I_k of the orders that are released in $[A^k, E^k)$ if there are no deliveries in I_k (as in $\hat{O}(k)$). The algorithm schedules one delivery at E^k; this incurs a delivery cost of D but reduces the total delay time over I_k by xS (with respect to $\hat{O}(k)$). Thus, $C_{ALG1}(I_k) = D+0.5S+\delta+D-xS \leq 2D+0.5S+\delta-2S = 2D - 1.5S + \delta$. (We used the inequality $x \geq 2$ shown above.) We also have $C_{OPT}(I_k) \geq Z(A^k, E^k + S) = D + 0.5S + \delta$, which completes the proof. □

In the following, for each of the categories *A*, *B*, *C*, we identify an exhaustive list of several situations that may happen for any basic interval I_k that belongs to this category. This will allow us later to show that the ratio for any block is not greater than $\frac{2D+0.5S}{D+0.5S}$, by analyzing how different situations for the basic intervals of a block can fit together within the block and using the logic outlined in the previous subsection.

Theorem 2. *If I_k belongs to Category A, then at least one of the following four situations happens.*

Situation A1. *I_k is end-incomplete, $C_{ALG1}(I_k) \leq 2D + 0.5S + \delta$, $C_{OPT}(I_k) \geq D + 0.5S + \delta$ for some $\delta \geq 0$.*
Situation A2. *I_k is end-complete, $C_{ALG1}(I_k) \leq 3D + \delta$, $C_{OPT}(I_k) \geq 2D + \delta$ for some $\delta \geq 0$.*
Situation A3. *I_k is end-complete, $C_{ALG1}(I_k) \leq D + 0.5S + \delta$, $C_{OPT}(I_k) \geq D + 0.5S + \delta$ for some $\delta \geq 0$.*
Situation A4. *I_k is end-complete, $C_{ALG1}(I_k) \leq 2D - 1.5S + \delta$, $C_{OPT}(I_k) \geq D + 0.5S + \delta$ for some $\delta \geq 0$.*

Proof. If I_k is end-incomplete, Lemma 4(a) and (c) implies that Situation A1 happens, taking into account $C_{OPT}(I_k) \geq Z(A^k, E^k + S) \geq D + 0.5S$ since I_k belongs to Category *A*. The remainder of the theorem follows from Lemma 5. □

Remark. If I_k belongs to Category *A*, the statement about $C_{ALG1}(I_k)$ and $C_{OPT}(I_k)$ in Situation A1 is valid also when I_k is end-complete, with the same proof. However, if I_k is end-complete, it would not be sufficient for our purposes; that is why we need Situations A2–A4.

Next, we state four theorems that will be proven jointly.

Theorem 3. *If I_k belongs to Category B, then at least one of the following four situations happens:*

Situation B1. *I_k is end-incomplete, $C_{ALG1}(I_k) \leq D + \delta$, $C_{OPT}(I_k) \geq 0.5D + \delta$ for some $\delta \geq 0$.*
Situation B2. *I_k is end-incomplete, $C_{ALG1}(I_k) \leq 2D - xS + \delta$, $C_{OPT}(I_k) \geq D - 0.5xS + \delta$ for some $\delta > 0$ and $x \geq 1$ such that $xS \leq D$.*
Situation B3. *I_k is end-complete, $C_{ALG1}(I_k) \leq D + \delta$, $C_{OPT}(I_k) \geq D + \delta$ for some $\delta \geq 0$.*
Situation B4. *I_k is end-complete, $C_{ALG1}(I_k) \leq 2D - xS + \delta$, $C_{OPT}(I_k) \geq D + \delta$ for some $\delta \geq 0$ and $x \geq 1$ such that $xS \leq D$.*

Theorem 4. *If I_k belongs to Category B and $D < S < 2D$, then either Situation* B3 *or Situation* B5 *happens, where Situation* B5 *is as follows:*

Situation B5. *I_k is end-incomplete, $C_{ALG1}(I_k) \leq D + \delta$, $C_{OPT}(I_k) \geq 0.5S + \delta$ for some $\delta \geq 0$.*

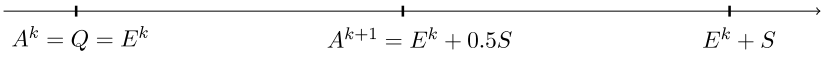

Fig. 2. Illustration for the proof of Theorems 3–6, Case 1.

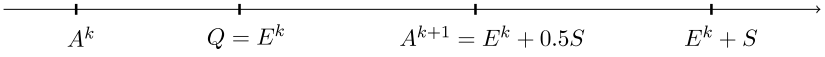

Fig. 3. Illustration for the proof of Theorems 3–6, Case 2.

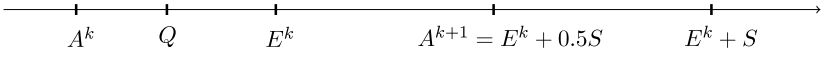

Fig. 4. Illustration for the proof of Theorems 3–6, Case 3.

Theorem 5. *If I_k belongs to Category C, then one of the following four situations happens:*

Situation C1. $C_{ALG1}(I_k) \le D + \delta$, $C_{OPT}(I_k) \ge 0.5D + 0.5S + \delta$ *for some* $\delta \ge 0$.
Situation C2. $C_{ALG1}(I_k) \le D + \delta$, $C_{OPT}(I_k) \ge D + \delta$ *for some* $\delta \ge 0$.
Situation C3. $C_{ALG1}(I_k) \le 2D - xS + \delta$, $C_{OPT}(I_k) \ge D + \delta$ *for some* $\delta \ge 0$ *and* $x \ge 1$ *such that* $xS \le D$.
Situation C4. $C_{ALG1}(I_k) \le 2D - xS + \delta$, $C_{OPT}(I_k) \ge D - 0.5xS + \frac{D}{x} - 0.5S + \delta$ *for some* $\delta \ge 0$, $x \ge 1$ *such that* $xS \le D$.

Theorem 6. *If I_k belongs to Category C and $D < S < 2D$, then either Situation* C2 *or Situation* C5 *happens, where Situation* C5 *is as follows:*

Situation C5. $C_{ALG1}(I_k) \le D + \delta$, $C_{OPT}(I_k) \ge S + \delta$ *for some* $\delta \ge 0$.

For the proof of Theorems 3–6, we need the following lemma.

Lemma 6. *For any a, b', b'' such that $a < b' \le b''$, if $O(a, b')$ has exactly one delivery that happens at instant τ' and $O(a, b'')$ has exactly one delivery that happens at instant τ'', then $\tau' \le \tau''$.*

Proof. See Appendix. □

Proof of Theorems 3–6. Suppose that I_k belongs to Category *B* or *C*. Since $Z(A^k, E^k + S) < D + 0.5S$ and $S < 2D$, we have that there is only one delivery in $O(A^k, E^k + S)$, which happens at some instant $Q \le E^k$. We need two auxiliary statements.

Auxiliary Statement 1. There are no order releases in (Q, A^{k+1}).

(*Proof*: Since $A^{k+1} = E^k + 0.5S$, if an order is released in (Q, A^{k+1}), the delay time of this order in $[A^k, E^k + S]$ is at least $0.5S$, which contradicts the assumption $Z(A^k, E^k + S) < D + 0.5S$.)

Auxiliary Statement 2. There is no delivery in (Q, A^{k+1}) in *OPT*.

(*Proof*: The statement follows from Auxiliary Statement 1 and the observation that deliveries in *OPT* can happen only at release times of orders.)

Let x be the number of orders released in $[A^k, Q]$, and let X be the total uninterrupted delay time in $[A^k, Q]$ of these orders. We need to consider three cases: (1) $E^k = A^k$; (2) $E^k > A^k$, and there is a delivery in $\hat{O}(k)$; (3) $E^k > A^k$, and there is no delivery in $\hat{O}(k)$. Each of the cases will have two subcases, depending on whether I_k belongs to Category *B* or *C*.

Case 1: $E^k = A^k$. Then, $Q = E^k = A^k$, $C_{ALG1}(I_k) = D$ (see Fig. 2). According to the definition, in this case x is the number of orders released at E^k, $x \ge 1$; then $S \ge \frac{D}{x}$ (otherwise $O(E^k, E^k + S)$ would not have a delivery at $Q = E^k$). Now there are two subcases.

Subcase 1-B: I_k belongs to Category *B*. If there is a delivery at E^k in *OPT*, then $C_{OPT}(I_k) \ge D$, and we have Situation B3 with $\delta = 0$. If there is no delivery at E^k in *OPT*, then, since there are no deliveries in (Q, A^{k+1}) in *OPT* according to Auxiliary Statement 2, the orders available at E^k in *OPT* accumulate total delay time at least $x \cdot 0.5S$ in $I_k = [E^k, E^k + 0.5S)$, so $C_{OPT}(I_k) \ge 0.5D$ (since $S \ge D/x$) and we have Situation B1 with $\delta = 0$ (if $S \ge D$, then we have Situation B5 with $\delta = 0$ since $C_{OPT}(I_k) \ge x \cdot 0.5S \ge 0.5S$).

Subcase 1-C: I_k belongs to Category *C*. If there is a delivery at E^k in *OPT*, then $C_{OPT}(I_k) \ge D$, and we have Situation C2 with $\delta = 0$. If there is no delivery at E^k in *OPT*, then, since there are no deliveries in (Q, A^{k+1}) in *OPT*, and since at least $(x + 1)$ orders are available at E^k in *OPT* (because I_k is start-incomplete), these orders accumulate total delay time at least $(x + 1) \cdot 0.5S$ in I_k, so $C_{OPT}(I_k) \ge 0.5D + 0.5S$ (since $S \ge D/x$) and we have Situation C1 with $\delta = 0$. (If $S \ge D$, then we have Situation C5 with $\delta = 0$ since $C_{OPT}(I_k) \ge (x + 1) \cdot 0.5S \ge S$.)

Case 2: $E^k > A^k$, and there is a delivery in $\hat{O}(k)$. Then, this delivery cannot be earlier than E^k (otherwise the algorithm would trigger at some $CT < E^k$), and it cannot be later than E^k because then the algorithm would not trigger at E^k (see Lemma 6); therefore, it must be at E^k. Then, $Q = E^k$ (see Lemma 6), see Fig. 3. (Observe that this also implies that there is at least one order release at E^k.)

As defined above, X is the total uninterrupted delay time in $[A^k, Q]$ of the orders from $J([A^k, Q])$; let Y be the total uninterrupted delay time of the same orders in $[Q, E^k + S]$. Then, $Y \geq D$; otherwise, there would be no delivery at Q in $O(A^k, E^k + S)$. Also note $Y \geq S$.

We have

$$C_{ALG1}(I_k) = X + D \tag{1}$$

and

$$C_{ALG1}(I_k) < D + 0.5S, \tag{2}$$

since $ALG1$ and $O(A^k, E^k + S)$ have delivery at the same time $Q = E^k$ in I_k and there are no order releases in $(E^k, E^k + 0.5S)$ and $Z(A^k, E^k + S) < D + 0.5S$ since I_k belongs to Category B or C. So, $X < 0.5S \leq D$. Now there are two subcases.

Subcase 2-B: I_k belongs to Category B. If OPT does not have deliveries in I_k, then $C_{OPT}(I_k) = X + 0.5Y$, since the orders from $J([A^k, E^k])$ would incur the total delay cost $0.5Y$ in $[E^k, E^k + 0.5S]$. Since $Y \geq D$ and $Y \geq S$, we have Situation B1 with $\delta = X$; see (1). If $S > D$, we have Situation B5 with $\delta = X$.

If OPT has a delivery at E^k, then there are no waiting orders at A^{k+1} in OPT. If this is the only delivery in I_k in OPT, then $C_{OPT}(I_k) = X + D$, and we have Situation B3 with $\delta = X$ (see (1)). If there is another delivery in I_k in OPT, then $C_{OPT}(I_k) \geq 2D$, and we again have Situation B3 since $C_{ALG1}(I_k) \leq D + 0.5S \leq 2D$ since $S \leq 2D$.

If OPT does not have a delivery at E^k but has a delivery in $[A^k, E^k)$, then there is at least one waiting order at A^{k+1} in OPT since there is at least one order release at E^k. An order released at $Q = E^k$ will incur delay cost $0.5S$ in $[E^k, E^k + 0.5S]$, and $C_{OPT}(I_k) \geq D + 0.5S$; we have Situation B1 with $\delta = 0.5S$ (see (2)). If $S > D$, then we have Situation B5 with $\delta = D$ because $C_{ALG1}(I_k) = X + D \leq 2D$.

Subcase 2-B is fully covered, because OPT does not have deliveries in (Q, A^{k+1}) (see Auxiliary Statement 2).

Subcase 2-C: I_k belongs to Category C. If OPT does not have deliveries in I_k, then $C_{OPT}(I_k) \geq X + 0.5Y + 0.5S$, since the jobs from $J([A^k, E^k])$ would incur the total delay cost $0.5Y$ in $[E^k, E^k + 0.5S]$ and an order waiting at A^k would incur a delay cost at least $0.5S$. Since $Y \geq D$, we have Situation C1 with $\delta = X$; (see (1)). If $S > D$, we have Situation C5 with $\delta = X$ since $Y \geq S$.

If OPT has a delivery at E^k, and this is the only delivery in I_k in OPT, then $C_{OPT}(I_k) \geq X + D$, and we have Situation C2 with $\delta = X$ (see (1)). If there is more than one delivery in I_k in OPT, then $C_{OPT}(I_k) \geq 2D$, and we again have Situation C2 since $C_{ALG1}(I_k) \leq D + 0.5S \leq 2D$ since $S \leq 2D$.

If OPT does not have a delivery at E^k but has a delivery in $[A^k, E^k)$, then an order released at $Q = E^k$ will incur delay cost $0.5S$ in $[E^k, E^k + 0.5S]$, and $C_{OPT}(I_k) \geq D + 0.5S$; we have Situation C2 with $\delta = 0.5S$ (see (2)).

Case 3: $E^k > A^k$, and there is no delivery in $\hat{O}(k)$ (see Fig. 4). This is possible only if $S \leq D$, because if $S > D$ then a delivery at E^k would improve the objective value for $\hat{O}(k)$, so this case is relevant only for Theorems 3 and 5. In this case, the total uninterrupted delay cost of orders from $J([A^k, Q])$ in the interval $(Q, E^k + S)$ is equal to D. (It cannot be less than D because then the delivery at Q in $O(A^k, E^k + S)$ would not be scheduled, and it cannot be greater than D because then $\hat{O}(k)$ would have the delivery.) Let $x \geq 1$ and X have the same meaning as before, and let R be the total uninterrupted delay cost of the orders from $J([A^k, Q])$ in the interval $[Q, E^k]$. We have $R = D - xS$, so $xS \leq D$. The algorithm initiates a delivery at E^k. We have

$$C_{ALG1}(I_k) = X + R + D = X + 2D - xS. \tag{3}$$

Note that $X + D \leq Z(A^k, E^k + S) \leq D + 0.5S$ because I_k belongs to Category B or C; therefore, $X \leq 0.5S$ and

$$C_{ALG1}(I_k) \leq 2D + 0.5S - xS. \tag{4}$$

Subcase 3-B: I_k belongs to Category B. According to Auxiliary Statement 2, OPT cannot have deliveries in (Q, A^{k+1}). If OPT has no delivery at Q but has a delivery in $[A^k, Q)$, then the order released at Q incurs a delay cost of at least $0.5S$ in I_k and will be waiting at A^{k+1} in OPT, so $C_{OPT}(I_k) \geq D + 0.5S$, which is covered by Situation B2 with $\delta = 0.5S$ (see (4)). If OPT has a delivery at Q, then it does not have deliveries in $[A^k, Q)$ (it cannot be beneficial since $O(A^k, E^k + S)$ does not have deliveries there and OPT does not have waiting orders at A^k), and $C_{OPT}(I_k) = X + D$, and we have Situation B4 with $\delta = X$ (see (3)). If OPT has no deliveries in I_k, then $C_{OPT}(I_k) = X + R + x \cdot 0.5S = X + D - xS + x \cdot 0.5S = X + D - 0.5xS$, and there is an order waiting at A^{k+1} in OPT, so we have Situation B2 with $\delta = X$ (as noted earlier, $xS \leq D$; see also (3)).

Subcase 3-C: I_k belongs to Category C. According to Auxiliary Statement 2, OPT cannot have deliveries in (Q, A^{k+1}). If OPT has no delivery at Q but has a delivery in $[A^k, Q)$, then the order released at Q incurs a delay cost of at least $0.5S$ in I_k, so $C_{OPT}(I_k) \geq D + 0.5S$, which is covered by Situation C3 with $\delta = 0.5S$ (see (4)). If OPT has more than one delivery in I_k, then $C_{OPT}(I_k) \geq 2D$, and we have Situation C2 with $\delta = D$ since $x \geq 1$ (see (4)). If OPT has a delivery at Q but no deliveries in $[A^k, Q)$, then $C_{OPT}(I_k) \geq X + D$, and we have Situation C3 with $\delta = X$ (see (3)). If OPT has no deliveries in I_k, then the order that is waiting at A^k has at least $\frac{D}{x} - 0.5S$ delay time since $E^k + S - Q \geq D/x$ (otherwise $O(A^k, E^k + S)$ would not have a delivery at Q) and therefore $A^{k+1} - A^k \geq \frac{D}{x} - 0.5S$. We have $C_{OPT}(I_k) \geq X + R + x \cdot 0.5S + \frac{D}{x} - 0.5S = X + D - xS + x \cdot 0.5S + \frac{D}{x} - 0.5S = X + D - 0.5xS + \frac{D}{x} - 0.5S$, so we have Situation C4 with $\delta = X$ (see (3)).

Theorems 3–6 have been proven. □

Lemma 7. *From Situations* A1–A4, B1–B5, C1–C5 *in Theorems 2–6, only in Situations* B1, B2, B5 *the ratio for* I_k *can be greater than* $\frac{2D+0.5S}{D+0.5S}$.

Proof. The statement of the lemma is straightforward for all situations except Situation C4, so we need to prove it for Situation C4. Suppose that Situation C4 holds but the ratio for I_k is greater than $\frac{2D+0.5S}{D+0.5S}$. Then,

$$\frac{2D - xS}{D - 0.5xS + \frac{D}{x} - 0.5S} > \frac{2D + 0.5S}{D + 0.5S},$$
$$\frac{2}{1 + \frac{1}{x}} > \frac{2D + 0.5S}{D + 0.5S}.$$

Note that $xS \le D$ implies $\frac{1}{x} \ge \frac{S}{D}$; we get

$$\frac{2D}{D + S} > \frac{2D + 0.5S}{D + 0.5S},$$

which is clearly wrong. Contradiction. □

Next we show that although the ratio for some basic intervals can be greater than $\frac{2D+0.5S}{D+0.5S}$, for any block the ratio cannot be greater than $\frac{2D+0.5S}{D+0.5S}$.

Theorem 7. *For any block B,*

$$\frac{C_{ALG1}(B)}{C_{OPT}(B)} \le \frac{2D + 0.5S}{D + 0.5S}. \tag{5}$$

Proof. If B is trivial and consists of only one basic interval I_k, then I_k is end-complete, and therefore Situations B1, B2, B5 cannot happen; these are the only situations where (5) may not hold according to Lemma 7; therefore, (5) holds.

Suppose that B is a nontrivial block, $B = I_k \cup \cdots \cup I_{k+p}$. From the basic intervals $I_k, \ldots, I_{k+p}$, only I_k can belong to Category B, because all other basic intervals are start-incomplete. Therefore, the ratios of the intervals $I_{k+1}, \ldots, I_{k+p}$ are not greater than $\frac{2D+0.5S}{D+0.5S}$, since only in Situations B1, B2, and B5 the ratio can be greater than $\frac{2D+0.5S}{D+0.5S}$.

Suppose that (5) does not hold; then

$$\frac{C_{ALG1}(I_k) + C_{ALG1}(I_{k+p})}{C_{OPT}(I_k) + C_{OPT}(I_{k+p})} > \frac{2D + 0.5S}{D + 0.5S}, \tag{6}$$

because the ratios for $I_{k+1}, \ldots, I_{k+p-1}$ are not greater than $\frac{2D+0.5S}{D+0.5S}$. We will show that this is not possible.

Note that since the ratio for I_{k+p} is not greater than $\frac{2D+0.5S}{D+0.5S}$, I_k must have ratio greater than $\frac{2D+0.5S}{D+0.5S}$, which is possible only in Situations B1, B2, or B5.

Case 1: $D < S < 2D$. Then, I_k corresponds to Situation B5, and we derive from (6)

$$\frac{D + C_{ALG1}(I_{k+p})}{0.5S + C_{OPT}(I_{k+p})} > \frac{2D + 0.5S}{D + 0.5S}. \tag{7}$$

The basic interval I_{k+p} must correspond to at least one of the Situations A2–A4, C2, C5 since I_{k+p} is end-complete and start-incomplete and thus cannot correspond to other situations (we used Theorems 2 and 6). Considering these situations one by one, we easily see that (7) is impossible; thus, we have a contradiction.

Case 2: $0 < S \le D$. Then, I_k corresponds to Situation B1 or B2, and in both situations, taking into account $S \le D$ and $x \ge 1$, (6) implies

$$\frac{2D - S + C_{ALG1}(I_{k+p})}{D - 0.5S + C_{OPT}(I_{k+p})} > \frac{2D + 0.5S}{D + 0.5S}. \tag{8}$$

I_{k+p} must correspond to at least one of the Situations A2–A4, C1–C4 since I_{k+p} is end-complete and start-incomplete and cannot correspond to any other situation. Considering Situations A2–A4, C1–C3 one by one, we easily see that (8) is impossible; however, this is not so obvious for Situation C4. For this situation, we need a detailed consideration.

Suppose that I_{k+p} corresponds to Situation C4. From (8), we derive

$$\frac{2D - S + 2D - xS}{D - 0.5S + D - 0.5xS + \frac{D}{x} - 0.5S} > \frac{2D + 0.5S}{D + 0.5S}. \tag{9}$$

We will show that the left side is not greater than $\frac{4D-2S}{2D-0.5S}$, which will be a contradiction with (9). Suppose the opposite:

$$\frac{4D-(x+1)S}{2D-S-0.5xS+\frac{D}{x}} > \frac{4D-2S}{2D-0.5S}. \tag{10}$$

Then,

$$8D^2-2(x+1)SD-2DS+0.5(x+1)S^2 > 8D^2-4DS-4DS+2S^2-2xDS+xS^2+4\frac{D^2}{x}-2\frac{DS}{x};$$

$$4DS-0.5xS^2-1.5S^2-4\frac{D}{x}(D-0.5S)>0.$$

Since $xS \le D$ and therefore $\frac{D}{x} \ge S$, we get

$$4DS-0.5xS^2-1.5S^2-4S(D-0.5S)>0;$$
$$0.5S^2(1-x)>0,$$

which is a contradiction since $x \ge 1$. This completes the proof. □

Corollary. *Algorithm 1 is ρ-competitive with $\rho = \frac{2D+0.5S}{D+0.5S}$.*

Now we are in a position to prove Theorem 1.

Proof of Theorem 1. Theorem 7 implies that $\frac{C_{ALG1}}{C_{OPT}} \le \frac{2D+0.5S}{D+0.5S}$, therefore the competitive ratio of Algorithm 1 cannot be greater than $\frac{2D+0.5S}{D+0.5S}$. To complete the proof of Theorem 1, it is sufficient to show that $\frac{C_{ALG1}}{C_{OPT}}$ can be made arbitrarily close to $\frac{2D+0.5S}{D+0.5S}$.

Case 1: $S \in (0, D)$. Consider the following instance. Orders are released at instants $a_1, b_1, c_1, a_2, b_2, c_2, \ldots, a_N, b_N, c_N$ ($3N$ orders altogether), where $a_1 = 0$, $a_{k+1} = c_k + 2\varepsilon$, $k \in [1 : N-1]$, $b_k = a_k + D - 0.5S$, $c_k = a_k + D - \varepsilon$, $k \in [1 : N]$, where ε is a small nonnegative real number. Algorithm 1 will schedule deliveries at instants $c_1, \ldots, c_N$ (it is triggered at instants $CT = a_k + D - S$, $k \in [1 : N]$, and Step 3 of Algorithm 1 is applicable), obtaining a solution of the total cost $C_{ALG1} = (2D + 0.5S - 2\varepsilon)N$. Consider the solution that has deliveries at $a_1, a_2, \ldots, a_N, c_N$; its total cost is $D(N+1) + (0.5S + 3\varepsilon)(N-1) + 0.5S - \varepsilon$. Therefore, $C_{OPT} \le D(N+1) + (0.5S + 3\varepsilon)N - 4\varepsilon$, and we have $\frac{C_{ALG1}}{C_{OPT}} \ge \frac{(2D+0.5S-2\varepsilon)N}{D(N+1)+(0.5S+3\varepsilon)N-4\varepsilon}$. With a choice of a sufficiently large N and a sufficiently small ε, this can be made arbitrarily close to $\frac{2D+0.5S}{D+0.5S}$.

Case 2: Suppose that $S \in [D, 2D)$. Consider the following instance. Orders are released at instants $a_1, b_1, c_1, a_2, b_2, c_2, \ldots, a_N, b_N, c_N$ ($3N$ orders altogether), where $a_1 = 0$, $a_{k+1} = c_k + 2\varepsilon$, $k \in [1 : N-1]$, $b_k = a_k + 0.5S$, $c_k = a_k + S - \varepsilon$, $k \in [1 : N]$. Algorithm 1 will schedule deliveries at instants $a_1, c_1, a_2, c_2, \ldots, a_N, c_N$ (it is triggered at instants $CT = a_k$, $k \in [1 : N]$), and Step 3 of Algorithm 1 is applicable), obtaining a solution of the total cost $C_{ALG1} = (2D + 0.5S - \varepsilon)N$. Consider the solution that has deliveries at $a_1, a_2, \ldots, a_N, c_N$; its total cost is $D(N+1) + (0.5S + 3\varepsilon)(N-1) + 0.5S - \varepsilon$. Therefore, $C_{OPT} \le D(N+1) + (0.5S + 3\varepsilon)N - 4\varepsilon$, and we have $\frac{C_{ALG1}}{C_{OPT}} \ge \frac{(2D+0.5S-\varepsilon)N}{D(N+1)+(0.5S+3\varepsilon)N-4\varepsilon}$. With a choice of a sufficiently large N and a sufficiently small ε, this can be made arbitrarily close to $\frac{2D+0.5S}{D+0.5S}$. □

3.4. Implementation issues and computational complexity

For Step 1 of Algorithm 1, we need to know $O(A^k, CT+S)$ (more specifically, the time of the first delivery in $O(A^k, CT+S)$ if there are any) for any instant $CT \in [A^k, E^k]$, which may create an impression that we need to be constantly re-solving problem $P(A^k, CT+S)$ as the current time CT changes. However, it is sufficient to solve $P(A^k, CT+S)$ no more than $2n$ times, namely, at instants CT when information about a new order release (at time $CT+S$) becomes available, and at initial instants A^k of the stages if there are already known undelivered orders. The idea is to solve problem $P(A^k, CT+S)$ "parametrically" each time, with A^k and S fixed and CT varied as a parameter, obtaining $O(A^k, CT+S)$ for a range of values CT where the set of known undelivered orders does not change. Below we discuss in more detail a possible way to implement this approach. Even though solving an instance of problem $P(a, b)$ has $O(n^2)$ computational complexity using the algorithm in the proof of Lemma 1, and we have to solve $O(n)$ such instances parametrically, the instances are related, and we will be able to organize computations in such a way that the total computational complexity is only $O(n^2 \log n)$.

Let $\tilde{P}_{i,j}$, $j \ge i$ denote Problem P where only the orders $o_i, o_{i+1}, \ldots, o_j$ are present, and let $\tilde{Z}_{i,j}$ denote its optimal objective value. Let $\tilde{\tau}(i, j)$ be the time of the first delivery in the lexicographically smallest optimal solution to $\tilde{P}_{i,j}$. Observe that $\tilde{Z}_{i,i} = D$ for any $i \in [1 : n]$. We also define $\tilde{Z}_{i,j} = 0$ and $\tilde{\tau}(i, j) = +\infty$ if $i > j$.

At any instant $CT \in [A^k, E^k]$ when there are some already known undelivered orders, we will keep auxiliary values $\tilde{Z}_{i',j}$, $\tilde{\tau}(i', j)$ for all j such that o_j is already known and undelivered (that is, $A^k \le r_j \le CT+S$), where i' is the index of the earliest known undelivered order. Observe that $o_{i'}$ is the first order released in $[A^k, +\infty)$. Also, we will keep values q_{ij} defined

in the proof of Lemma 1 in the Appendix, for all i, j, $i < j$ such that the orders o_i, o_j are already known and undelivered. Each necessary value $\tilde{Z}_{i',j}$, $i' < j$ is obtained at time $\max\{A^k, r_j - S\}$, which is the earliest instant of Stage k when the order o_j is known, using the recursive relation

$$\tilde{Z}_{i',j} = \min_{p\in[i'-1:j-1]} \{\tilde{Z}_{i',p} + q_{pj}\} \tag{11}$$

(recall that $\tilde{Z}_{i',i'-1} = 0$ and $\tilde{Z}_{i',i'} = D$). If more than one value $\tilde{Z}_{i',j}$ needs to be obtained at the same time (which may happen if $A^k \geq r_j - S$ for several values of j), they are obtained using (11) in the order of increasing j. To justify (11), observe that $\tilde{Z}_{i',p} + q_{pj}$ is the optimal objective value for $\tilde{P}_{i',j}$ under the condition that the last delivery before r_j is at r_p if $p \geq i'$, and that there are no deliveries at all before r_j if $p = i' - 1$. Value $\tilde{\tau}(i', j)$ is obtained at the same time $\max\{A^k, r_j - S\}$; if p^* is the minimizer in (11), then $\tilde{\tau}(i', j) = \tilde{\tau}(i', p^*)$ if $p^* \geq i'$, and $\tilde{\tau}(i', j) = r_j$ if $p^* = i' - 1$. If there is more than one minimizer in (11), then we choose one that corresponds to the smallest value $\tilde{\tau}(i', p^*)$.

The overall computational complexity of obtaining all values q_{ij} is $O(n^2)$. Let us estimate the total computational complexity of obtaining all necessary values $\tilde{Z}_{i',j}$ and $\tilde{\tau}(i', j)$ using (11) throughout Algorithm 1. Observe that if at a Stage k deliveries are scheduled in Step 3, then all orders released in $[A^k, E^k + S)$ get delivered at this stage. If at Stage k deliveries are scheduled in Step 2, then all orders released in $[A^k, E^k + 0.5S)$ get delivered at this stage, and all orders released in $[E^k + 0.5S, E^k + S)$ will be delivered at the next Stage $k + 1$ since the length of each basic interval is at least $0.5S$. Therefore, any r_j can belong to $[A^k, E^k + S)$ for at most two values of k. This implies that for any j, we need to obtain at most two values $\tilde{Z}_{i',j}$ using (11) throughout Algorithm 1, and overall we need to obtain at most $2n$ values $\tilde{Z}_{i',j}$ using (11). Thus, the overall computational complexity of computing all necessary values $\tilde{Z}_{i',j}$ and $\tilde{\tau}(i', j)$ is $O(n^2)$.

For any a, b, $0 \leq a \leq b$, let $\tau(a, b)$ be the time of the first delivery in $O(a, b)$; if there are no deliveries in $O(a, b)$, then $\tau(a, b) = +\infty$. For Step 1 of Algorithm 1, we need to know $\tau(A^k, CT + S)$ at any instant $CT \in [A^k, E^k]$ of Stage k.

Suppose that at an instant $CT \in [A^k, E^k]$, $o_{i'}, o_{i'+1}, \ldots, o_{i''}$ are the currently known undelivered orders, that is, all the orders released in $[A^k, CT + S]$. Consider the function

$$\Psi(b) = \min_{p\in[i'-1:i'']} \Phi^{(p)}(b), \tag{12}$$

where for any $p \in [i' - 1 : i'']$

$$\Phi^{(p)}(b) = \tilde{Z}_{i',p} + \sum_{j=p+1}^{i''} (b - r_j).$$

Observation 3. $\Psi(b) = Z(A^k, b)$ for any $b \in [r_{i''}, r_{i''+1})$.

To verify Observation 3, observe that for any $b \in [r_{i''}, r_{i''+1})$, $\Phi^{(p)}(b)$ is the optimal objective value for $P(A^k, b)$ under the condition that the last delivery before b is at r_p if $p \geq i'$, and that there are no deliveries in $[A^k, b)$ if $p = i' - 1$. (Recall that $\tilde{Z}_{i',i'-1} = 0$ and $\tilde{Z}_{i',i'} = D$.)

For any b, let $p^*(b)$ be the minimizer in (12); in case of a tie when there are several minimizers, we take one that has the smallest value $\tilde{\tau}(i', p^*(b))$. Observe that for any $b \in [r_{i''}, r_{i''+1})$,

$$\tau(A^k, b) = \tilde{\tau}(i', p^*(b)). \tag{13}$$

Function $\Psi(b)$ is the lower envelope of $i'' - i' + 2 \leq n + 1$ linear functions $\Phi^{(p)}(b)$, $p \in [i' - 1 : i'']$. Therefore, it is continuous piece-wise linear with at most n breakpoints; $p^*(b)$ is piece-wise constant with the same breakpoints. According to (13), $\tau(A^k, b)$ can change only at the breakpoints of $\Psi(b)$ when b varies in $[r_{i''}, r_{i''+1})$. Thus, if we have the breakpoints of $\Psi(b)$ and the values $p^*(b)$ that correspond to the intervals between the consecutive breakpoints, we will have $\tau(A^k, b)$ for any $b \in [r_{i''}, r_{i''+1})$.

At the instant $\max\{A^k, r_{i''} - S\}$, which is the earliest instant of Stage k when the order $o_{i''}$ is already known, we obtain an explicit representation of functions $\Psi(b)$ and $p^*(b)$ on $[r_{i''}, +\infty)$. (We consider $\Psi(b)$ and $p^*(b)$ on $[r_{i''}, +\infty)$ although we need them only on $[r_{i''}, r_{i''+1})$, because at time $\max\{A^k, r_{i''} - S\}$ we do not know $r_{i''+1}$ yet.) The computational complexity is $O(n \log n)$, since the lower envelope of n linear functions can be obtained in $O(n \log n)$ time [14] and all values $\tilde{Z}_{i',p}$, $p \in [i' - 1 : i'']$ are known at the instant $\max\{A^k, r_{i''} - S\}$. According to (13), this also gives us $\tau(A^k, CT + S)$ for all instants of Stage k when $o_{i''}$ is already known but $o_{i''+1}$ is not known yet. (Observe that $A^k < r_{i''+1} - S$, because we assumed that at some instant of Stage k only the orders $o_{i'}, \ldots, o_{i''}$ are known.)

When a new order becomes known, or a new stage is started, functions $\Psi(b)$ and $p^*(b)$ are re-computed. Since there are at most $n + 1$ stages and there are no order releases during the last stage, functions $\Psi(b)$ and $p^*(b)$ should be re-computed at most $2n$ times. Since the computational complexity of each re-computation is $O(n \log n)$, the total computational complexity of updating functions $\Psi(b)$ and $p^*(b)$ in Algorithm 1 is $O(n^2 \log n)$.

The computational complexity of solving problems $P'(k)$ at Step 3 throughout Algorithm 1 is $O(n^2)$, because problems $P'(k)$ of different stages involve non-overlapping subsets of the set of orders $\{o_1, \ldots, o_n\}$, and for problem $P'(k)$ with m orders the computational complexity is $O(m^2)$.

We conclude that Algorithm 1 can be implemented with computational complexity $O(n^2 \log n)$.

Remark. We note that the computational complexity $O(n^2 \log n)$ of Algorithm 1 is higher than the complexity $O(n^2)$ of the algorithm in the proof of Lemma 1 for the off-line version, and than the complexity $O(n)$ of the on-line algorithm from [4] for the on-line version. This is natural, because the semi-online problem has a more complicated information structure than both the off-line and the on-line versions, which are the limiting cases of the semi-online problem when $S \to \infty$ and $S \to 0$, respectively.

4. The case $S \geq 2D$

Suppose that $S \geq 2D$. The following algorithm is proposed in this case for Problem $P^{(S)}$.

Algorithm 2. Initially, set $k = 1$. Value k will be interpreted as a stage number.

1. Wait until the first order release; suppose this happens at time τ_k.
2. Consider the interval $[\tau_k, \tau_k + S]$; all order release times in this interval are already known.

Case 1: There is an order release instant τ'_k in $[\tau_k, \tau_k + S]$ such that $\tau'_k + D < \tau_k + S$ and there are no order releases in $(\tau'_k, \tau'_k + D]$. (In other words, there is a subinterval of $[\tau_k, \tau_k + S]$ of length greater than D without order releases.) If there is more than one such τ'_k, take the latest of them. Solve problem $P(\tau_k, \tau'_k)$ (defined in Section 2); as before, $Z(\tau_k, \tau'_k)$ and $O(\tau_k, \tau'_k)$ are its optimal objective value and the lexicographically smallest optimal solution. Schedule deliveries in $[\tau_k, \tau'_k]$ according to $O(\tau_k, \tau'_k)$, and add a delivery at τ'_k. Wait until the first order release after τ'_k, set τ_{k+1} to be equal to the time of this order release, and set $k := k + 1$. Return to 2.

Case 2: $[\tau_k, \tau_k + S]$ does not have a subinterval of length greater than D without order releases. Let τ'_k be the last order release instant in $[\tau_k, \tau_k + S]$. Solve problem $P(\tau_k, \tau'_k)$, schedule deliveries in $[\tau_k, \tau'_k]$ according to $O(\tau_k, \tau'_k)$, and add a delivery at τ'_k. Wait until the first order release after τ'_k, set τ_{k+1} to be equal to the time of this order release, and set $k := k + 1$. Return to 2.

The description of Algorithm 2 is completed.

Let *ALG*2 be the solution obtained by Algorithm 2, and let C_{ALG2} be the total cost of *ALG*2. For any interval I, let $C_{ALG2}(I)$ be the total cost of *ALG*2 over I (defined similarly to $C_{ALG1}(I)$ and $C_{OPT}(I)$).

Theorem 8. $\frac{C_{ALG2}}{C_{OPT}} \leq \frac{S+D}{S}$.

Proof. Let $\tau''_k = \tau'_k + D$ if Case 1 happened in Stage k, and $\tau''_k = \tau_k + S$ if Case 2 happened in Stage k. Observe that intervals $[\tau_k, \tau''_k]$, $k = 1, 2, \ldots$ do not overlap, and no cost is incurred outside of these intervals in *ALG*2. To prove the theorem, it is sufficient to prove that

$$\frac{C_{ALG2}([\tau_k, \tau''_k])}{C_{OPT}([\tau_k, \tau''_k])} \leq \frac{S+D}{S} \quad \text{for any } k.$$

If Case 1 happened in Stage k, then there must be a delivery at τ'_k in *OPT*, and we have

$$C_{OPT}([\tau_k, \tau''_k]) \geq Z(\tau_k, \tau'_k) + D = C_{ALG2}([\tau_k, \tau''_k]), \quad \text{so} \quad \frac{C_{ALG2}([\tau_k, \tau''_k])}{C_{OPT}([\tau_k, \tau''_k])} \leq 1.$$

If Case 2 happened in Stage k, then the total cost incurred in $[\tau_k, \tau''_k]$ in any solution is at least S. We have $C_{ALG2}([\tau_k, \tau''_k]) = Z(\tau_k, \tau'_k) + D \leq Z(\tau_k, \tau''_k) + D$, $C_{OPT}([\tau_k, \tau''_k]) \geq Z(\tau_k, \tau''_k)$, and $Z(\tau_k, \tau''_k) \geq S$, thus

$$\frac{C_{ALG2}([\tau_k, \tau''_k])}{C_{OPT}([\tau_k, \tau''_k])} \leq \frac{Z(\tau_k, \tau''_k) + D}{Z(\tau_k, \tau''_k)} \leq \frac{S+D}{S}. \quad \square$$

Remark. The theorem holds when $S > D$, but we need it only for the case $S \geq 2D$ which is considered in this section.

Theorem 9. *The competitive ratio of Algorithm 2 is* $\frac{D+S}{S}$.

Proof. Theorem 8 implies that the competitive ratio cannot be greater than $\frac{D+S}{S}$. To complete the proof, it is sufficient to show that $\frac{C_{ALG2}}{C_{OPT}}$ can be made arbitrarily close to $\frac{D+S}{S}$.

Let $k = \lceil \frac{S}{D} \rceil$. Consider the following instance. Orders are released at instants $a_1^{(0)}, a_1^{(1)}, \ldots, a_1^{(k)}, a_2^{(0)}, a_2^{(1)}, \ldots, a_2^{(k)}, \ldots, a_N^{(0)}, a_N^{(1)}, \ldots, a_N^{(k)}$ ($N(k+1)$ orders altogether), where $a_1^{(0)} = 0$, $a_{i+1}^{(0)} = a_i^{(k)} + \varepsilon$ for any $i \in [1 : N-1]$ where ε is a small nonnegative real number, and for any $i \in [1 : N]$:

(a) If S is not a multiple of D, then

$$a_i^{(1)} = a_i^{(0)} + D, \qquad a_i^{(2)} = a_i^{(1)} + \left(\frac{S}{D} - \left\lfloor \frac{S}{D} \right\rfloor\right) D, \qquad a_i^{(j+1)} = a_i^{(j)} + D \quad \text{for all } j \in [2 : k-1]$$

(observe that $k \geq 3$ in this case since $S \geq 2D$ in this section);

(b) If S is a multiple of D, then

$$a_i^{(j+1)} = a_i^{(j)} + D \quad \text{for all } j \in [0 : k-1].$$

If S is not a multiple of D, Algorithm 2 will schedule deliveries at all order release instants except $a_i^{(1)}$, $i \in [1 : N]$ (Case 2 is applicable), obtaining the total cost of $N(S + D)$. (Remember that $O(\tau_k, \tau_k')$ is the lexicographically smallest optimal solution to $P(\tau_k, \tau_k')$.) Consider the solution that has deliveries at all order release instants except $a_i^{(1)}$, $i \in [1 : N]$, and $a_i^{(k)}$, $i \in [1 : N-1]$; its total cost is $NS + (N-1)\varepsilon + D$. Therefore, $C_{OPT} \leq NS + (N-1)\varepsilon + D$, and we have $\frac{C_{ALG2}}{C_{OPT}} \geq \frac{N(S+D)}{NS+(N-1)\varepsilon+D}$. With a choice of a sufficiently large N and a sufficiently small ε, this can be made arbitrarily close to $\frac{S+D}{S}$.

If S is a multiple of D, Algorithm 2 will schedule deliveries at all order release instants (Case 2 is applicable), obtaining the total cost of $N(S + D)$. Consider the solution that has deliveries at all order release instants except $a_i^{(k)}$, $i \in [1 : N-1]$; its total cost is $NS + (N-1)\varepsilon + D$. Using the same argument as in the previous case, we see that $\frac{C_{ALG2}}{C_{OPT}}$ can be made arbitrarily close to $\frac{S+D}{S}$ with a proper choice of N and ε. □

Remark. The computational complexity of Algorithm 2 is $O(n^2)$, because problems $P(\tau_k, \tau_k')$ of different stages involve non-overlapping subsets of the set of orders $\{o_1, \ldots, o_n\}$.

5. A lower bound on the competitive ratio of any semi-online algorithm

In this section, we obtain a lower bound on the competitive ratio of *any* deterministic semi-online algorithm for Problem $P^{(S)}$.

Theorem 10. *No deterministic semi-online algorithm for* Problem $P^{(S)}$ *can have competitive ratio smaller than* $\frac{2D}{D+S}$.

Proof. We will assume $S < D$ because otherwise the statement of the theorem is obvious. Suppose we have a deterministic semi-online algorithm for Problem $P^{(S)}$ that we will call Algorithm B. The optimal objective value for Problem $P^{(S)}$ will be denoted C_{OPT}, and the objective value of the solution obtained by Algorithm B will be denoted C_{ALGB}. We will demonstrate that there is an instance of Problem $P^{(S)}$ such that for this instance $\frac{C_{ALGB}}{C_{OPT}} \geq \frac{2D}{D+S} - \delta$, where $\delta > 0$ can be made arbitrarily small. We obtain such an instance using the notion of an "adversary" [5] that makes decisions to release orders based on actions of Algorithm B. Due to the semi-online context, when making a decision on releasing an order at an instant τ, the adversary can use only the information about actions of Algorithm B prior to the instant $\tau - S$, because actions of Algorithm B after $\tau - S$ may depend on the information about the order release at τ. In other words, the adversary watches the actions of Algorithm B and makes decisions based on these actions, but at any time t the adversary can make decisions only about order releases after time $t + S$.

The adversary releases the first order at some time r_1, and waits until Algorithm B makes a delivery. When the delivery occurs at a time d_1, the adversary makes a decision to release the next order at time $r_2 = d_1 + S + \varepsilon$, for some $\varepsilon > 0$ whose choice will be discussed later, and again waits until Algorithm B makes a delivery. The process continues in the same way: the adversary releases an order $S + \varepsilon$ time units after each delivery, i.e. at times $r_i = d_{i-1} + S + \varepsilon$, until $2N + 1$ orders have been released and delivered, for some integer $N > 1$ whose choice will be discussed later.

Observe that for the obtained instance, $C_{ALGB} = (2N + 1)D + (d_{2N+1} - r_1) - 2N(S + \varepsilon)$. Now let us obtain an upper bound on C_{OPT}. Consider the following two off-line solutions for the obtained instance:

Solution 1. Deliveries are at times $r_1, r_3, r_5, \ldots, r_{2k+1}, \ldots, r_{2N+1}$ (i.e. at the release times of the odd-numbered orders).

Solution 2. Deliveries are at times $r_2, r_4, r_6, \ldots, r_{2k}, \ldots, r_{2N}, r_{2N+1}$ (i.e. at the release times of the even-numbered orders and the last order).

Observe that all deliveries except the first delivery in Solution 1 and the last delivery in Solution 2 deliver two orders. Let C_i be the objective value of Solution i, $i = 1, 2$. We get

$$C_1 = (N + 1)D + \sum_{k=1}^{N} (r_{2k+1} - r_{2k}), \qquad C_2 = (N + 1)D + \sum_{k=1}^{N} (r_{2k} - r_{2k-1}).$$

Note that $C_1+C_2 = 2(N+1)D+r_{2N+1}-r_1 \le 2(N+1)D+d_{2N+1}-r_1$. This implies that $\min\{C_1, C_2\} \le (N+1)D+0.5(d_{2N+1}-r_1)$. Taking into account $C_{OPT} \le \min\{C_1, C_2\}$, we get

$$\begin{aligned}\frac{C_{ALGB}}{C_{OPT}} &\ge \frac{(2N+1)D + (d_{2N+1}-r_1) - 2N(S+\varepsilon)}{(N+1)D + 0.5(d_{2N+1}-r_1)} \\ &= \frac{2ND + (d_{2N+1}-r_1-2N(S+\varepsilon)) + D}{ND + 0.5(d_{2N+1}-r_1-2N(S+\varepsilon)) + N(S+\varepsilon) + D} \\ &\ge \frac{2D + D/N}{D+S+\varepsilon+D/N}.\end{aligned}$$

In the last inequality, we used $d_{2N+1} - r_1 \ge 2N(S+\varepsilon)$. Thus, for any $\delta > 0$, values N and ε can be chosen to ensure $\frac{C_{ALGB}}{C_{OPT}} \ge \frac{2D}{D+S} - \delta$. Therefore, the competitive ratio of Algorithm B cannot be smaller than $\frac{2D}{D+S}$. □

6. Final remarks

The results of the paper would also be applicable in the situation where orders have non-zero and possibly unequal processing times, no more than one order can be processed simultaneously, and preemption is allowed, if at any time τ we know the orders that will be released in $[\tau, \tau+S]$. It is known that the optimal off-line and on-line policy with respect to processing the orders is to use the shortest remaining processing time (SRPT) rule [4]. Processing orders according to the SRPT rule defines the times when the orders are available for delivery, which can be considered as new release times for the algorithms of this paper.

The lower bound on the competitive ratio of any deterministic semi-online algorithm for Problem $P^{(S)}$ obtained in Section 5 is smaller than the competitive ratio of our algorithm. Thus, it would be interesting to try to obtain stronger lower bounds or find a semi-online algorithm with a better competitive ratio than our algorithm. This may be a direction for future research.

Acknowledgment

This work was supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) to the first author.

Appendix

Proof of Lemma 1. In any optimal solution to Problem P, the last delivery is at time r_n, and all deliveries are at release times of some orders. Define values q_{ij}, $i \in [0:n-1]$, $j \in [1:n]$, $j > i$, as follows: $q_{ij} = \sum_{k=i+1}^{j}(r_j - r_k) + D$. Value q_{ij} represents the sum of delay times of orders $o_{i+1}, \ldots, o_j$ assuming there is a delivery at r_j and no deliveries in $[r_{i+1}, r_j)$, plus the cost of one delivery (at r_j). All values q_{ij} can be obtained in $O(n^2)$ total time. For any $j \in [1:n]$, let P_j denote Problem P where only the orders $o_j, \ldots, o_n$ are present, and let Z_j be its optimal objective value. We also define $Z_{n+1} = 0$. Clearly, Z_1 is the optimal objective value for Problem P. All values $Z_j, j = 1, \ldots, n$ can be found in $O(n^2)$ total time by standard dynamic programming using the following recursive relations:

$$Z_j = \min_{i \in [j+1:n+1]} \{q_{(j-1)(i-1)} + Z_i\}, \quad j \in [1:n]. \tag{14}$$

To justify (14), observe that $q_{(j-1)(i-1)} + Z_i$ is the optimal objective value for P_j under the condition that the first delivery is at r_{i-1}. Consider the following algorithm that, given all values q_{ij} and Z_j, finds an optimal solution to Problem P.

Algorithm A.

1. Set $i = 0$, $Z^* = Z_1$.
2. Find $j' = \min\{j \mid j > i \text{ and } q_{ij} + Z_{j+1} = Z^*\}$. Schedule a delivery at $r_{j'}$.
3. If $j' = n$, STOP and output the obtained solution.

   Otherwise, set $Z^* = Z_{j'+1}$, $i = j'$, and go to 2.

It is clear that the algorithm obtains the optimal solution to Problem P which is lexicographically smallest. Algorithm A takes $O(n)$ time when all values q_{ij} and Z_j are known, thus the overall complexity is $O(n^2)$. This dynamic programming procedure is a version of standard reaching algorithms for shortest path problems in acyclic networks (e.g., [1]); Problem P can be viewed as a shortest path problem from r_0 to r_n in a directed network with nodes $r_0, r_1, \ldots, r_n$ and edges (r_i, r_j), $i < j$ of length q_{ij}. □

Proof of Lemma 6. Suppose that $\tau' > \tau''$. This cannot happen if $b' = b''$ since $O(a, b')$ is the lexicographically smallest solution, so we can assume that $b' < b''$. Also, τ' cannot be equal to b' since this cannot be optimal for problem $P(a, b')$, therefore we have $a < \tau'' < \tau' < b' < b''$. Let W_1 be the number of orders released in $[a, \tau'']$, and W_2 be the number of orders released in $(\tau'', \tau']$. Since there must be an order release at τ' and τ'', we have $W_1 > 0$ and $W_2 > 0$.

Let Δ'' be the change in the total cost for the problem $P(a, b'')$ if we replace the delivery at τ'' with a delivery at τ'. Clearly, $\Delta'' = W_1(\tau' - \tau'') - W_2(b'' - \tau')$. Since the delivery at τ'' is optimal for $P(a, b'')$, we get $\Delta'' \geq 0$ and

$$W_1(\tau' - \tau'') - W_2(b'' - \tau') \geq 0. \tag{15}$$

Let Δ' be the change in the total cost for the problem $P(a, b')$ if we replace the delivery at τ' with a delivery at τ''. Clearly $\Delta = -\big(W_1(\tau' - \tau'') - W_2(b' - \tau')\big)$. Using (15) and $b'' > b'$, we have $\Delta' < 0$. Therefore, a single delivery at τ'' is better than a single delivery at τ' for $P(a, b')$. Contradiction. □

References

[1] R. Ahuja, T. Magnanti, J. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice-Hall, Upper Saddle River, New Jersey, 1993.
[2] I. Averbakh, On-line integrated production–distribution scheduling problems with capacitated deliveries, European Journal of Operational Research 200 (2) (2010) 377–384.
[3] I. Averbakh, M. Baysan, Semi-online two-level supply chain scheduling problems, Journal of Scheduling 15 (3) (2012) 381–390.
[4] I. Averbakh, Z. Xue, On-line supply chain scheduling problems with preemption, European Journal of Operational Research 181 (2007) 500–504.
[5] A. Borodin, R. El-Yaniv, Online Computation and Competitive Analysis, Cambridge University Press, 1998.
[6] Z.-L. Chen, Integrated production and outbound distribution scheduling: review and extensions, Operations Research 58 (1) (2010) 130–148.
[7] Z.-L. Chen, N. Hall, Supply chain scheduling: conflict and cooperation in assembly systems, Operations Research 55 (2007) 1072–1089.
[8] R. Fu, T. Ji, J. Yuan, Y. Lin, On-line scheduling in a parallel batch processing system to minimize makespan using restarts, Theoretical Computer Science 374 (2007) 196–202.
[9] N.G. Hall, C.N. Potts, Supply chain scheduling: batching and delivery, Operations Research 51 (4) (2003) 566–584.
[10] N.G. Hall, C.N. Potts, The coordination of scheduling and batch deliveries, Annals of Operations Research 135 (2005) 41–64.
[11] J.A. Hoogeveen, A.P.A. Vestjens, A best possible on-line algorithm for minimizing maximum delivery time on a single machine, SIAM Journal on Discrete Mathematics 13 (1) (2000) 56–63.
[12] K. Pruhs, J. Sgall, E. Torng, On-line scheduling, in: Joseph Y.-T. Leung (Ed.), Handbook of Scheduling: Algorithms, Models, and Performance Analysis, CRC Press, 2004, pp. 15.1–15.41 (Chapter 15).
[13] S. Seiden, Randomized on-line scheduling with delivery times, Journal of Combinatorial Optimization 3 (1999) 399–416.
[14] M. Sharir, P. Agarwal, Davenport-Schinzel Sequences and their Geometric Applications, Cambridge University Press, New York, 1995.
[15] J. Tian, R. Fu, J. Yuan, On-line scheduling with delivery time on a single batch machine, Theoretical Computer Science 374 (2007) 49–57.
[16] M. van den Akker, H. Hoogeveen, N. Vakhania, Restarts can help in the on-line minimization of the maximum delivery time on a single machine, Journal of Scheduling 3 (2000) 333–341.